FIG. 1

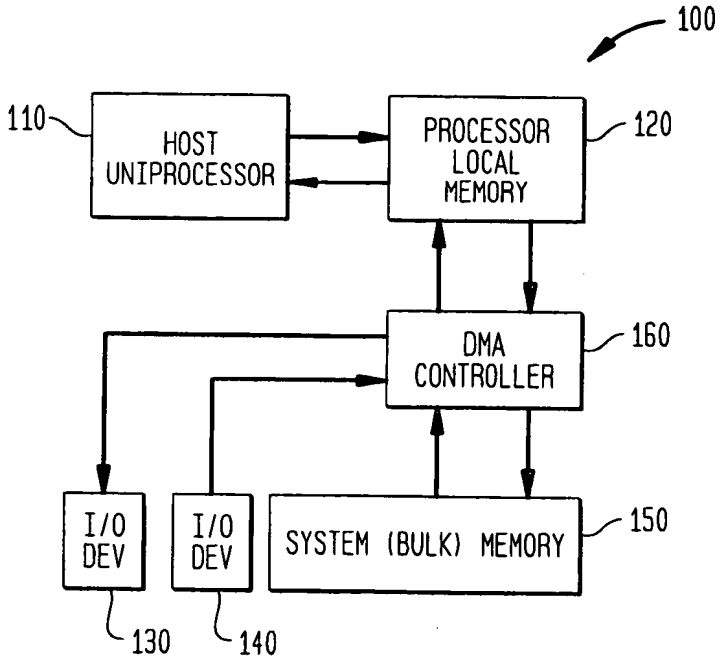
100
110
HOST UNIPROCESSOR
PROCESSOR LOCAL MEMORY
120
DMA CONTROLLER
160
I/O DEV
I/O DEV
SYSTEM (BULK) MEMORY
150
130
140

FIG. 2

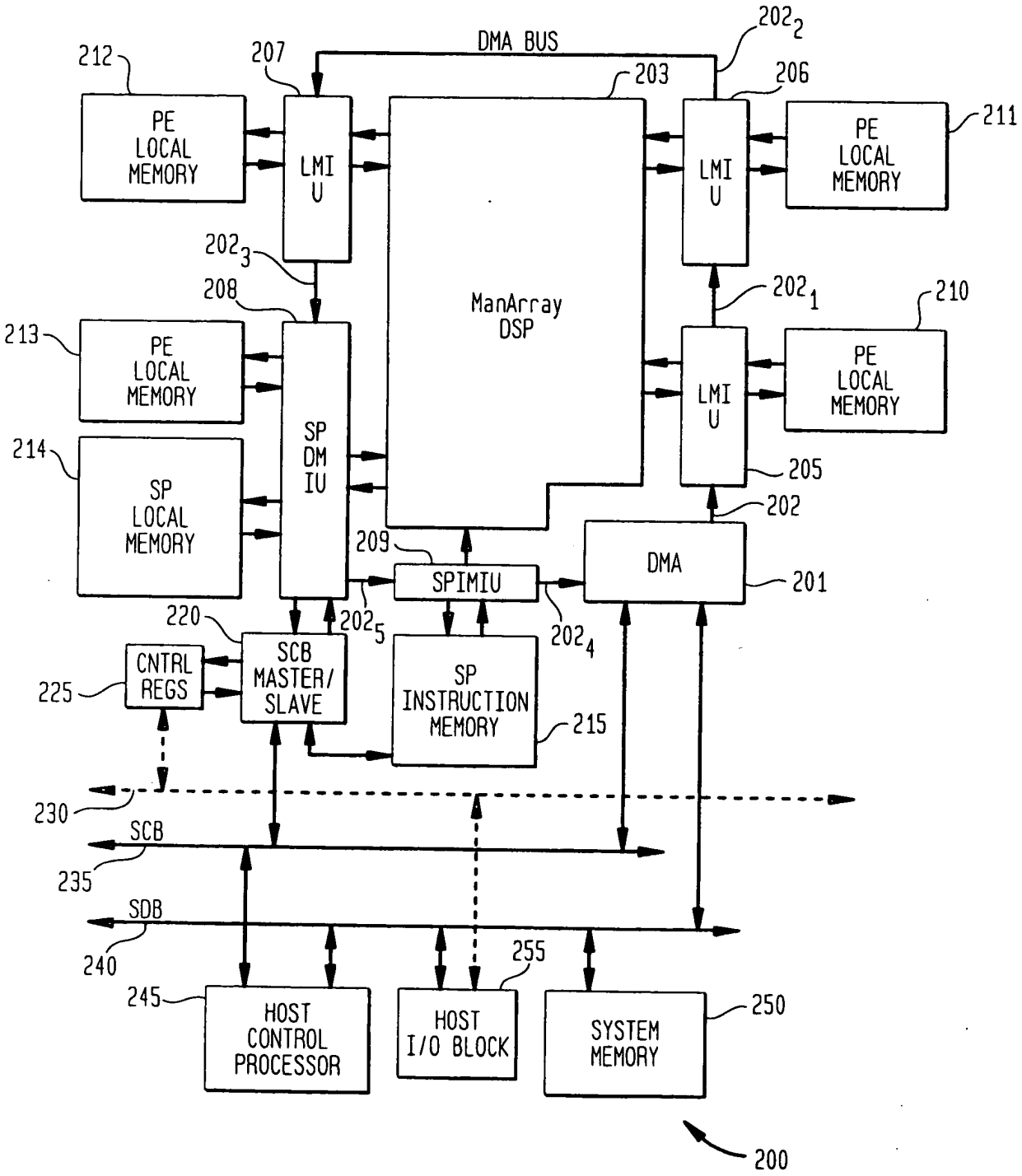
DMA BUS
202_2
212
207
203
206
PE LOCAL MEMORY
LMI U
ManArray DSP
LMI U
PE LOCAL MEMORY
211
202_3
208
202_1
210
213
PE LOCAL MEMORY
SP DM IU
LMI U
PE LOCAL MEMORY
214
SP LOCAL MEMORY
205
202
209
SPIMIU
DMA
201
220
202_5
202_4
CNTRL REGS
225
SCB MASTER/ SLAVE
SP INSTRUCTION MEMORY
215
230
SCB
235
SDB
240
255
245
HOST CONTROL PROCESSOR
HOST I/O BLOCK
SYSTEM MEMORY
250
200

FIG. 3

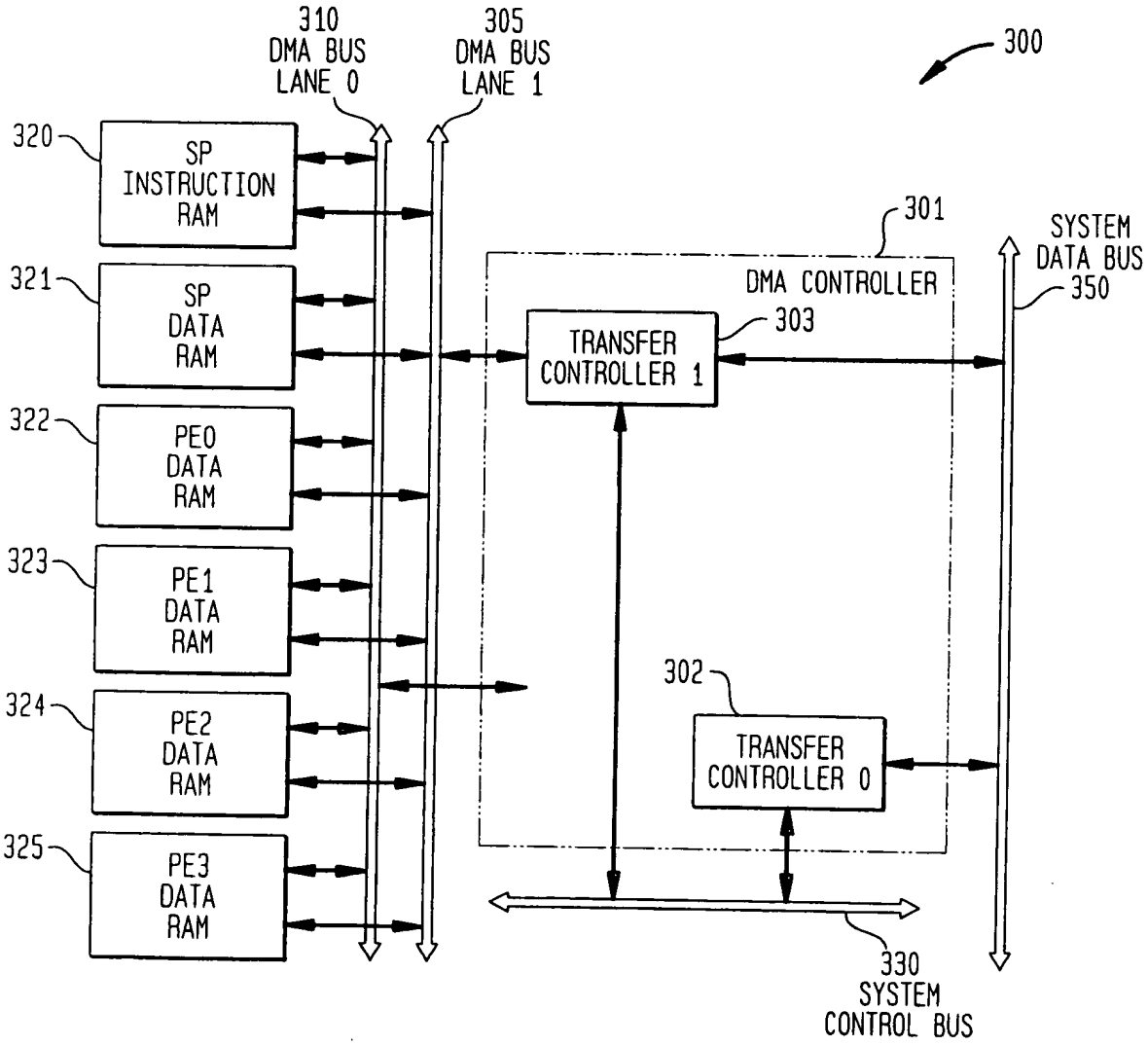
310
DMA BUS
LANE 0
305
DMA BUS
LANE 1
300
320
SP
INSTRUCTION
RAM
321
SP
DATA
RAM
322
PE0
DATA
RAM
323
PE1
DATA
RAM
324
PE2
DATA
RAM
325
PE3
DATA
RAM
301
DMA CONTROLLER
303
TRANSFER
CONTROLLER 1
302
TRANSFER
CONTROLLER 0
SYSTEM
DATA BUS
350
330
SYSTEM
CONTROL BUS

FIG. 4

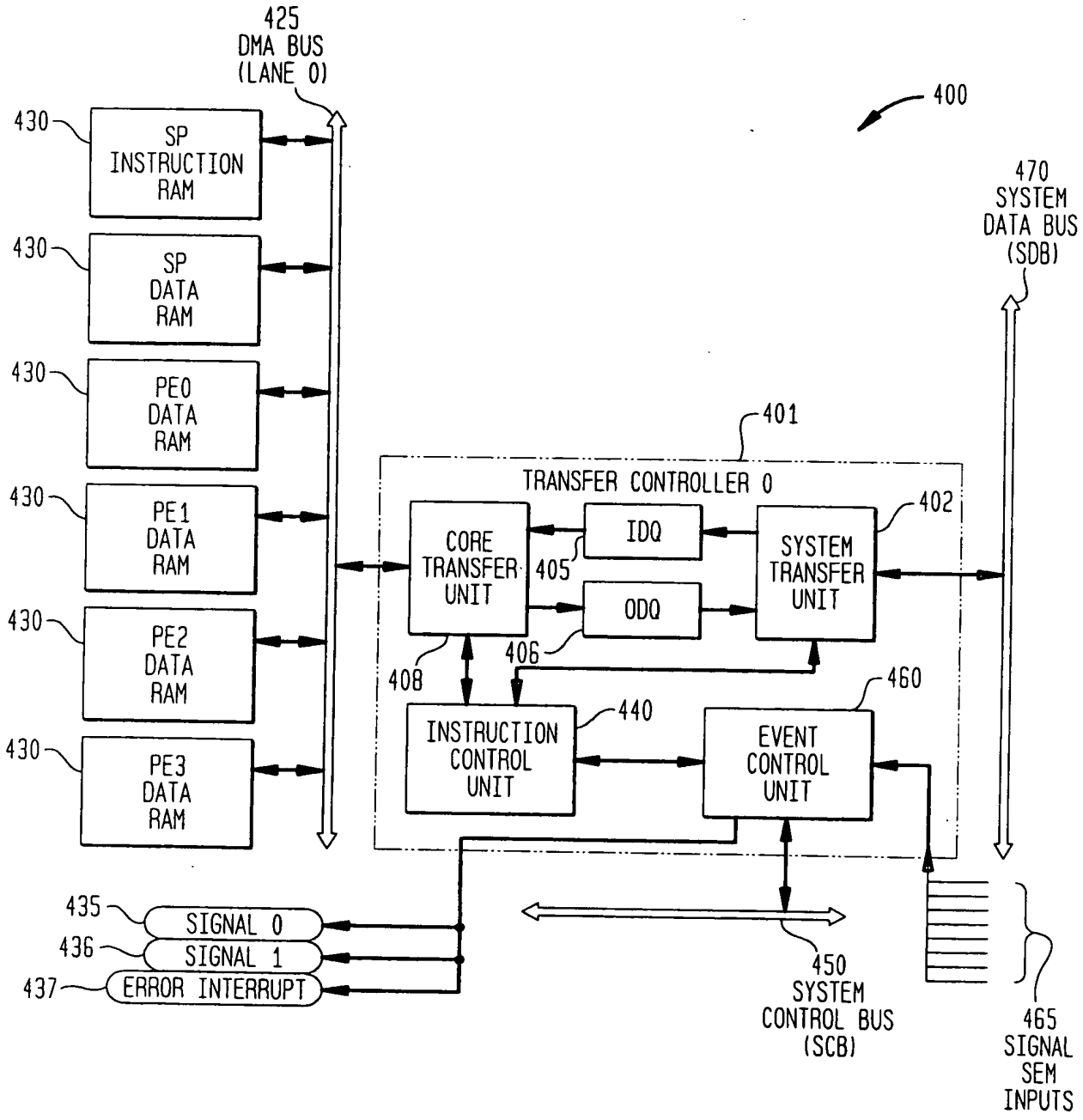
425
DMA BUS
(LANE 0)
400
430
SP
INSTRUCTION
RAM
430
SP
DATA
RAM
430
PE0
DATA
RAM
430
PE1
DATA
RAM
430
PE2
DATA
RAM
430
PE3
DATA
RAM
470
SYSTEM
DATA BUS
(SDB)
401
TRANSFER CONTROLLER 0
CORE
TRANSFER
UNIT
405
IDQ
ODQ
SYSTEM
TRANSFER
UNIT
402
406
408
460
440
INSTRUCTION
CONTROL
UNIT
EVENT
CONTROL
UNIT
435
SIGNAL 0
436
SIGNAL 1
437
ERROR INTERRUPT
450
SYSTEM
CONTROL BUS
(SCB)
465
SIGNAL
SEM
INPUTS

FIG. 5

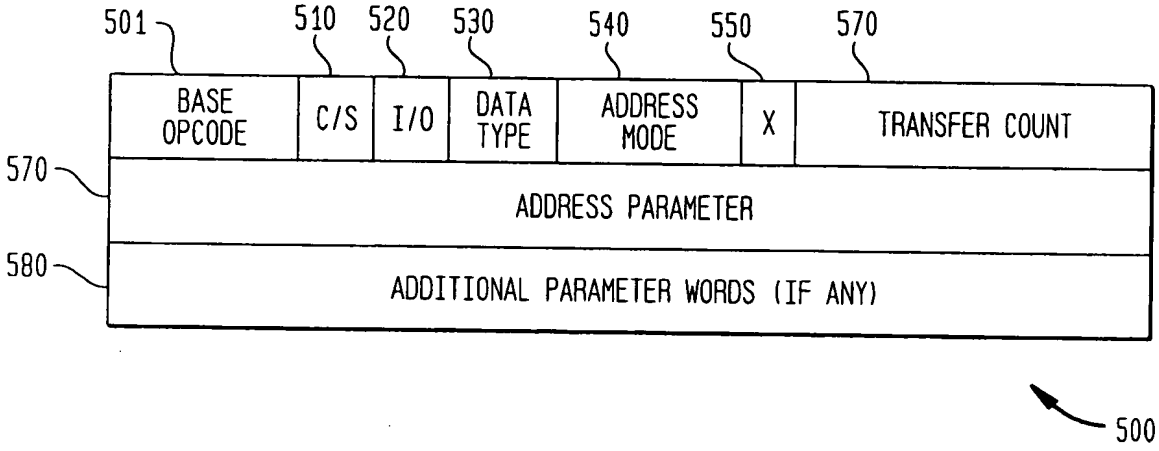
501
510
520
530
540
550
570
BASE OPCODE
C/S
I/O
DATA TYPE
ADDRESS MODE
X
TRANSFER COUNT
570
ADDRESS PARAMETER
580
ADDITIONAL PARAMETER WORDS (IF ANY)
500

FIG. 6

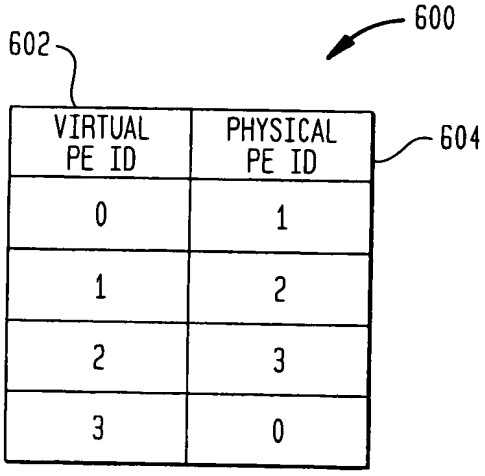
600
602
VIRTUAL PE ID
PHYSICAL PE ID
604
0
1
1
2
2
3
3
0

FIG. 7

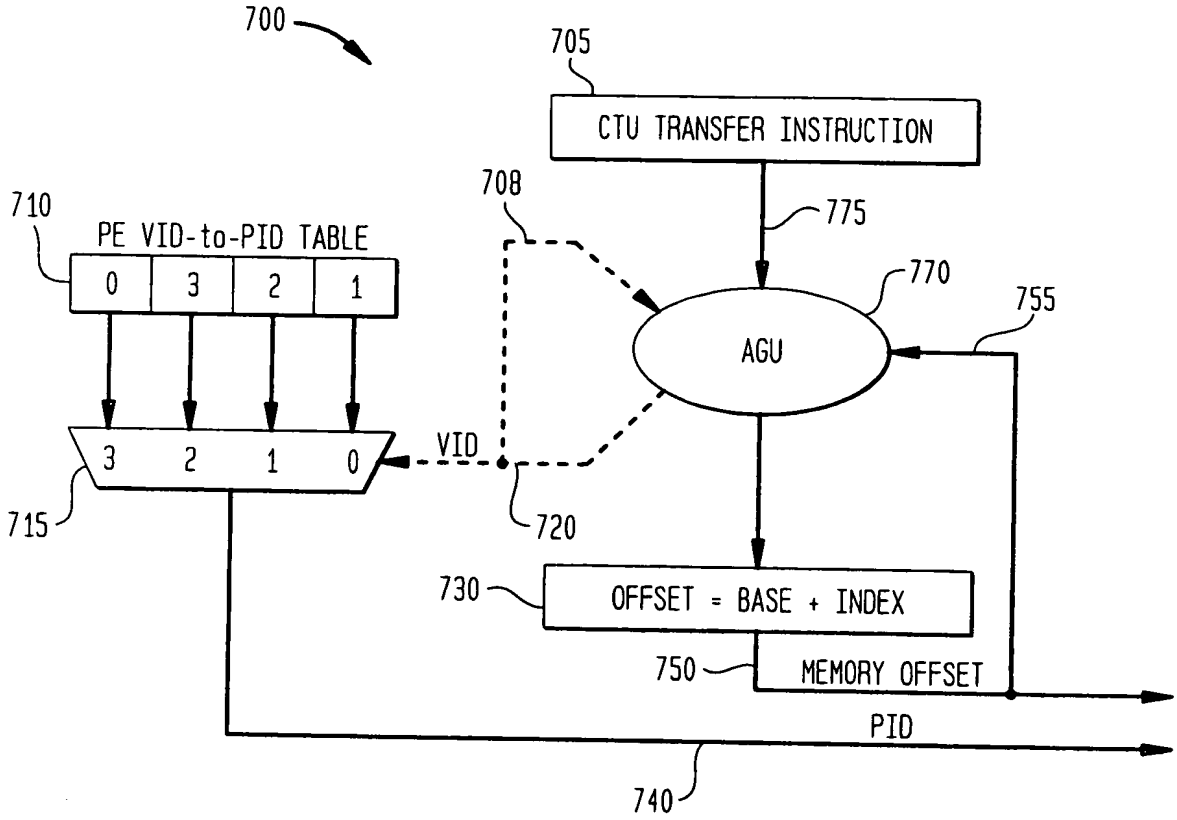
700
705
CTU TRANSFER INSTRUCTION
708
775
710
PE VID-to-PID TABLE
0
3
2
1
770
755
AGU
3
2
1
0
VID
715
720
730
OFFSET = BASE + INDEX
750
MEMORY OFFSET
PID
740

FIG. 8

800

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
00		0110				MA TYPE 01		(USED FOR 2x4 TRANSLATE TABLE)																2x2 TABLE							
(USED FOR 4x4 TRANSLATE TABLE)																															

2x2 TABLE	CONTAINS A TABLE OF TWO BIT PE IDs. A SEQUENCE OF TWO BIT VALUES (STARTING WITH 0) WHICH SPECIFY THE PE VID, ARE APPLIED AS AN INDICES INTO THIS TABLE WHEN ONE OF THE PE ADDRESSING MODES IS USED IN A TRANSFER INSTRUCTION. THE TRANSLATED VALUE IS THEN USED TO PERFORM THE MEMORY ACCESS. WITH THIS APPROACH, PEs MAY BE ACCESSED IN ANY ORDER FOR THESE MODES.
MA TYPE	ManArray TYPE SPECIFIES THE CONFIGURATION TARGETED AND THEREFORE THE SIZE OF THE TABLE. 00 - 1x2 (UP TO 2 PEs) 01 - 2x2 (UP TO 4 PEs) 10 - 2x4 (UP TO 8 PEs) 11 - 4x4 (UP TO 16 PEs)

FIG. 9

900

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
USED FOR PE ID TRANSLATION TABLES LARGER THAN 4 ELEMENTS																								PID3		PID2		PID1		PID0	

FIG. 10

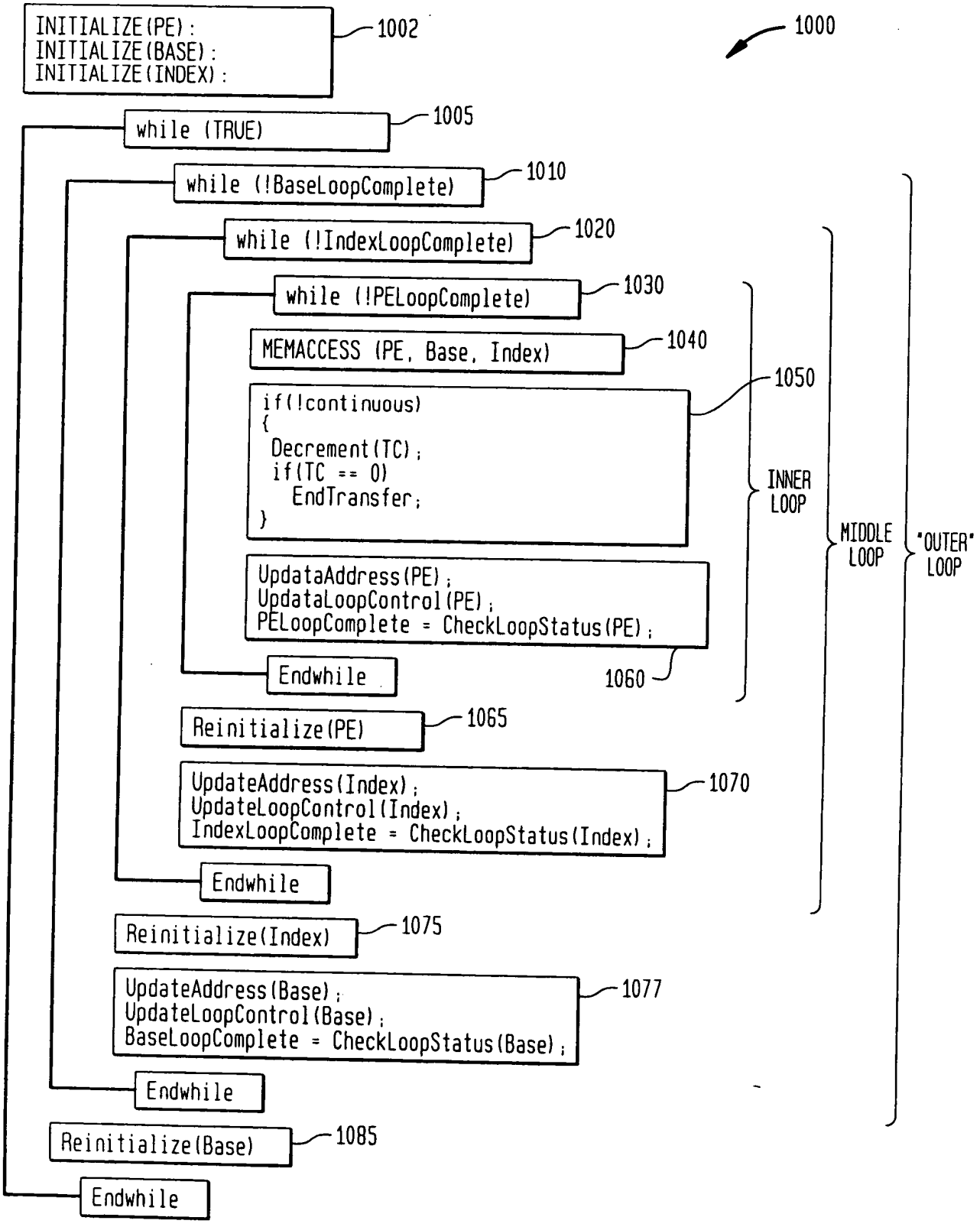
INITIALIZE(PE):
INITIALIZE(BASE):
INITIALIZE(INDEX):
1002
1000
while (TRUE)
1005
while (!BaseLoopComplete)
1010
while (!IndexLoopComplete)
1020
while (!PELoopComplete)
1030
MEMACCESS (PE, Base, Index)
1040
if(!continuous)
{
Decrement(TC);
if(TC == 0)
EndTransfer;
}
1050
UpdataAddress(PE);
UpdataLoopControl(PE);
PELoopComplete = CheckLoopStatus(PE);
Endwhile
1060
INNER LOOP
Reinitialize(PE)
1065
UpdateAddress(Index);
UpdateLoopControl(Index);
IndexLoopComplete = CheckLoopStatus(Index);
1070
Endwhile
MIDDLE LOOP
Reinitialize(Index)
1075
UpdateAddress(Base);
UpdateLoopControl(Base);
BaseLoopComplete = CheckLoopStatus(Base);
1077
Endwhile
Reinitialize(Base)
1085
Endwhile
"OUTER" LOOP

FIG. 11

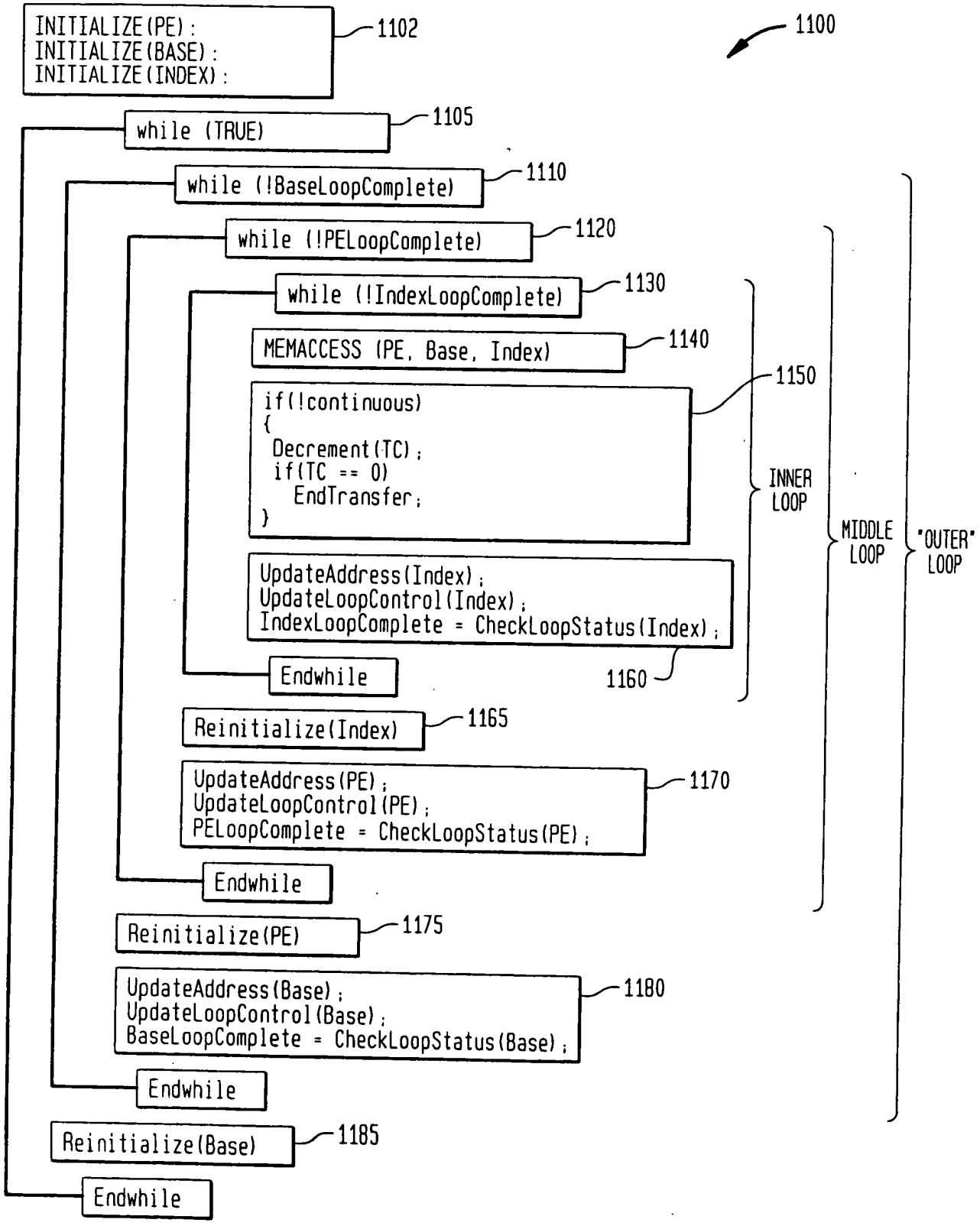
INITIALIZE(PE):
INITIALIZE(BASE):
INITIALIZE(INDEX):
1102
1100
while (TRUE)
1105
while (!BaseLoopComplete)
1110
while (!PELoopComplete)
1120
while (!IndexLoopComplete)
1130
MEMACCESS (PE, Base, Index)
1140
if(!continuous)
{
Decrement(TC);
if(TC == 0)
EndTransfer;
}
1150
UpdateAddress(Index);
UpdateLoopControl(Index);
IndexLoopComplete = CheckLoopStatus(Index);
Endwhile
1160
INNER LOOP
Reinitialize(Index)
1165
UpdateAddress(PE);
UpdateLoopControl(PE);
PELoopComplete = CheckLoopStatus(PE);
1170
Endwhile
Reinitialize(PE)
1175
UpdateAddress(Base);
UpdateLoopControl(Base);
BaseLoopComplete = CheckLoopStatus(Base);
1180
Endwhile
Reinitialize(Base)
1185
Endwhile
MIDDLE LOOP
"OUTER" LOOP

FIG. 12

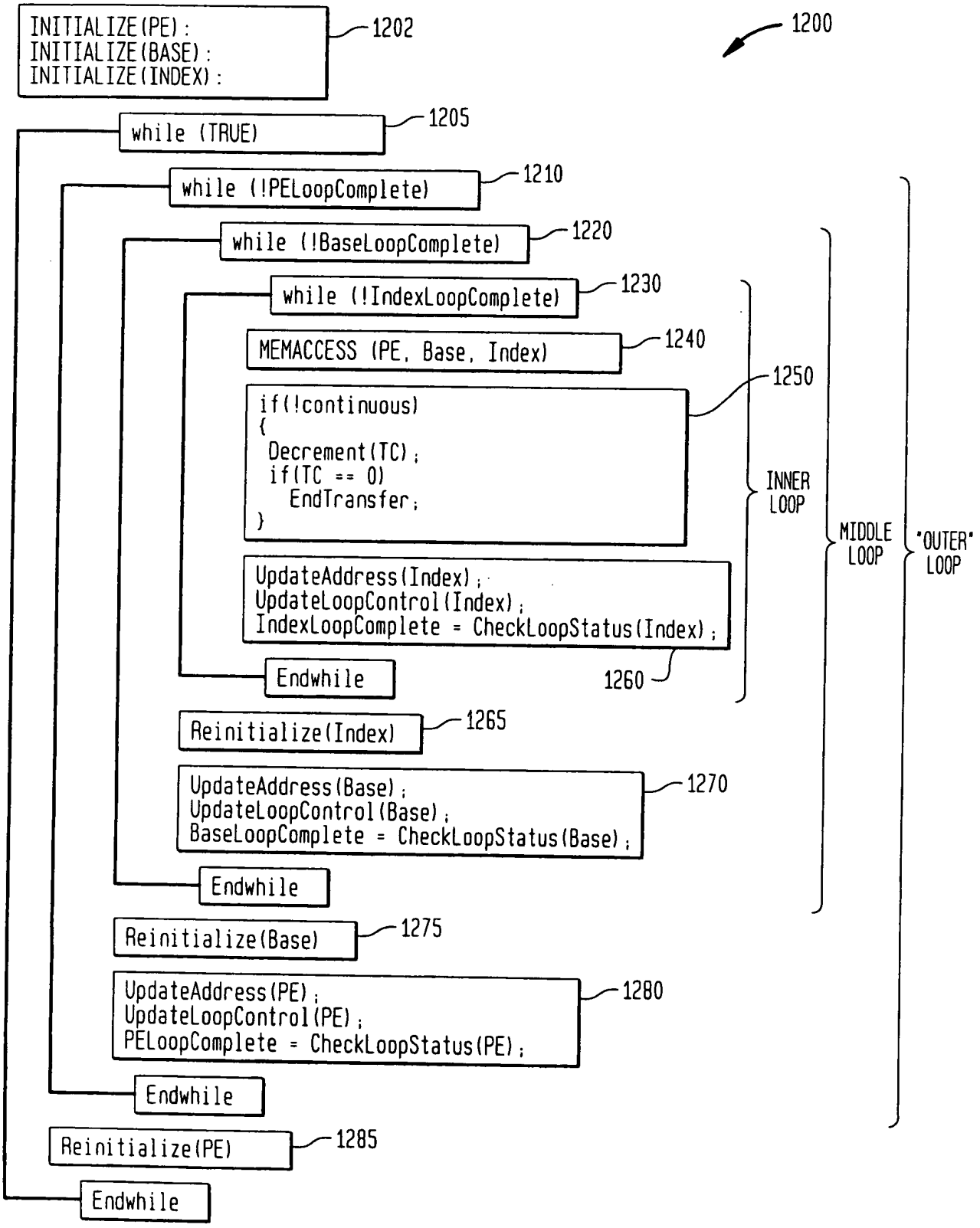
INITIALIZE(PE):
INITIALIZE(BASE):
INITIALIZE(INDEX):
1202
1200
while (TRUE)
1205
while (!PELoopComplete)
1210
while (!BaseLoopComplete)
1220
while (!IndexLoopComplete)
1230
MEMACCESS (PE, Base, Index)
1240
if(!continuous)
{
Decrement(TC);
if(TC == 0)
EndTransfer;
}
1250
UpdateAddress(Index);
UpdateLoopControl(Index);
IndexLoopComplete = CheckLoopStatus(Index);
Endwhile
1260
INNER LOOP
Reinitialize(Index)
1265
UpdateAddress(Base);
UpdateLoopControl(Base);
BaseLoopComplete = CheckLoopStatus(Base);
1270
Endwhile
MIDDLE LOOP
Reinitialize(Base)
1275
UpdateAddress(PE);
UpdateLoopControl(PE);
PELoopComplete = CheckLoopStatus(PE);
1280
Endwhile
"OUTER" LOOP
Reinitialize(PE)
1285
Endwhile

FIG. 13

1300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CTU TRANSFER					I/O	TYPE		BLOCKCYCLIC				X	RSVD			CORE TRANSFER COUNT (CTC)															
RESERVED																STARTING TRANSFER ADDRESS (WITHIN PE MEMORY)															
LOOP CTRL			PE COUNT			BASE UPDATE COUNT RANGE: 1 TO 256										BASE UPDATE (STRIDE) RANGE:															
INDEX COUNT (HOLD) RANGE:																RESERVED												INDEX UPDATE RANGE: 1-256			

LOOP CTRL	LOOP CTRL SPECIFIES A PARTICULAR ORDER IN WHICH PE, BASE AND INDEX VALUES ARE UPDATED. THREE POSSIBLE ORDERS ARE SELECTABLE WHICH CORRESPOND TO THREE ASSIGNMENTS OF PE, BASE AND INDEX UPDATE TO THREE NESTED CONTROL LOOPS (OUTER, MIDDLE AND INNER). 00 - BASE (OUTER), INDEX (MIDDLE), PE (INNER) - BIP 01 - BASE (OUTER), PE (MIDDLE), INDEX (INNER) - BPI 10 - PE (OUTER), BASE (MIDDLE), INDEX (INNER) - PBI
PE COUNT	SPECIFIES THE NUMBER OF PEs TO BE ACCESSED FOR EACH TIME THE PE COUNTER IS SIGNALED TO RELOAD. VALID VALUES ARE: 0000 - MAX NUMBER OF PEs AS SPECIFIED IN THE PE CONFIGURATION REGISTER 0001 - 1 0010 - 2 0011 - 3 ETC., ETC.
BASE UPDATE (STRIDE)	DISTANCE BETWEEN SUCCESSIVE BLOCKS. UNITS ARE OF "DATA TYPE" SIZE.
BASE UPDATE COUNT	USED FOR PBI LOOP CONTROL. SPECIFIES THE NUMBER OF TIMES THE BASE IS UPDATED BEFORE EXITING TO THE OUTER LOOP (PE UPDATE). RANGE IS 1 TO 256.
INDEX COUNT (HOLD)	NUMBER OF CONTIGUOUS DATA ITEMS IN A BLOCK
INDEX UPDATE	DISTANCE BETWEEN SUCCESSIVE ITEMS WITHIN A BLOCK. UNITS ARE OF "TYPE" SIZE.

FIG. 14

1400

LOOP CONTROL: BIP (PE ID VARIES FIRST, THEN INDEX, THEN BASE)				
ADDRESS	PE0	PE1	PE2	PE3
0x0000	0	1	2	3
0x0001				
0x0002	4	5	6	7
0x0003				
0x0004				
0x0005				
0x0006				
0x0007				
0x0008	8	9	10	11
0x0009				
0x000a	12	13	14	15

- AN INBOUND SEQUENCE OF 16 DATA ELEMENTS WITH VALUES 0,1,2,3,...15
- PETABLE SETTING OF 0x000000E4 (NO TRANSLATION OF PE IDs)
- TSI.block INSTRUCTION IN THE STU (READING THE 16 VALUES FROM SYSTEM MEMORY)
- TCI.blockcyclic INSTRUCTION IN THE CTU WITH PE COUNT = 4, LOOP CONTROL = BIP, BASE UPDATE = 8, BASE COUNT =, INDEX UPDATE = 2, INDEX COUNT = 2

FIG. 15

1500

LOOP CONTROL: BPI (INDEX VARIES FIRST, THEN PE ID, THEN BASE)				
ADDRESS	PE0	PE1	PE2	PE3
0x0000	0	2	4	6
0x0001				
0x0002	1	3	5	7
0x0003				
0x0004				
0x0005				
0x0006				
0x0007				
0x0008	8	10	12	14
0x0009				
0x000a	9	11	13	15

- AN INBOUND SEQUENCE OF 16 DATA ELEMENTS WITH VALUES 0,1,2,3,...15
- PETABLE SETTING OF 0x000000E4 (NO TRANSLATION OF PE IDs)
- TSI.block INSTRUCTION IN THE STU (READING THE 16 VALUES FROM SYSTEM MEMORY)
- TCI.blockcyclic INSTRUCTION IN THE CTU WITH PE COUNT = 4, LOOP CONTROL = BPI, BASE UPDATE = 8, BASE COUNT =, INDEX UPDATE = 2, INDEX COUNT = 2

FIG. 16

1600

LOOP CONTROL: PBI (INDEX VARIES FIRST, THEN BASE, THEN PE ID)

ADDRESS	PE0	PE1	PE2	PE3
0x0000	0	4	8	12
0x0001				
0x0002	1	5	9	13
0x0003				
0x0004				
0x0005				
0x0006				
0x0007				
0x0008	2	6	10	14
0x0009				
0x000a	3	7	11	15

- AN INBOUND SEQUENCE OF 16 DATA ELEMENTS WITH VALUES 0,1,2,3,...15
- PETABLE SETTING OF 0x000000E4 (NO TRANSLATION OF PE IDs)
- TSI.block INSTRUCTION IN THE STU (READING THE 16 VALUES FROM SYSTEM MEMORY)
- TCI.blockcyclic INSTRUCTION IN THE CTU WITH PE COUNT = 4, LOOP CONTROL = BPI, BASE UPDATE = 8, BASE COUNT =, INDEX UPDATE = 2, INDEX COUNT = 2

NOTE THAT A FOR PBI MODE, THE BASE COUNT MUST BE 2 IN ORDER TO GET 2 "BLOCKS" OF DATA. INDEX COUNT CORRESPONDES TO THE NUMBER OF ELEMENTS WRITTEN BEFORE UPDATING THE NEXT ADDRESS VARIABLE. THE GAP BETWEEN ELEMENTS WITHIN A PE IS DUE TO THE INDEX UPDATE VALUE OF 2 (RATHER THAN 1)

FIG. 17

1700

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CTU TRANSFER					I/O	TYPE		PE SELECT INDEX				X	RSVD			CORE TRANSFER COUNT (CTC)															
INDEX COUNT					RESERVED											STARTING TRANSFER ADDRESS (WITHIN PE MEMORY)															
LOOP CTRL					INDEX COUNT							BASE UPDATE COUNT				BASE UPDATE (STRIDE)															
IU7					IU6							IU5				IU4				IU3				IU2				IU1				IU0	

LOOP CTRL	LOOP CTRL SPECIFIES A PARTICULAR ORDER IN WHICH PE, BASE AND INDEX VALUES ARE UPDATED. THREE POSSIBLE ORDERS ARE SELECTABLE WHICH CORRESPOND TO THREE ASSIGNMENTS OF PE, BASE AND INDEX UPDATE TO THREE NESTED CONTROL LOOPS (OUTER, MIDDLE AND INNER). 00 - BASE (OUTER), INDEX (MIDDLE), PE (INNER) - BIP 01 - BASE (OUTER), PE (MIDDLE), INDEX (INNER) - BPI 10 - PE (OUTER), BASE (MIDDLE), INDEX (INNER) - PBI
PE COUNT	SPECIFIES THE NUMBER OF PEs TO BE ACCESSED FOR EACH TIME THE PE COUNTER IS SIGNALED TO RELOAD. VALID VALUES ARE: 0000 - MAX NUMBER OF PEs AS SPECIFIED IN THE PE CONFIGURATION REGISTER 0001 - 1 0010 - 2 0011 - 3 ETC., ETC.
BASE UPDATE (STRIDE)	DISTANCE BETWEEN SUCCESSIVE BLOCKS. UNITS ARE OF "DATA TYPE" SIZE.
BASE UPDATE COUNT	USED FOR PBI LOOP CONTROL. SPECIFIES THE NUMBER OF TIMES THE BASE IS UPDATED BEFORE EXITING TO THE OUTER LOOP (PE UPDATE). RANGE IS 1 TO 256.
IUx	IU0 - IU7 FORM AN INDEX UPDATE TABLE WITH EACH ENTRY BEING A 4-BIT UPDATE VALUE. UPDATE VALUES ARE INTEGERS IN THE RANGE OF -8 TO +7
INDEX COUNT	NUMBER OF TIMES TO EXECUTE THE INDEX UPDATE LOOP. THIS VARIABLE PROVIDES THE LOOP EXIT CONTROL FOR THE INDEX LOOP.

FIG. 18

1800

LOOP CONTROL: BIP (INDEX VARIES FIRST, THEN BASE, THEN PE ID)

ADDRESS	PE0	PE1	PE2	PE3
0x0000	0	1	2	3
0x0001	24	25	26	27
0x0002	4	5	6	7
0x0003	20	21	22	23
0x0004	8	9	10	11
0x0005	16	17	18	19
0x0006	12	13	14	15
0x0007				
0x0008	28	29	30	31
0x0009				
0x000a	32	33	34	35

PATTERN ABOVE RESULTS FROM AFTER A TRANSFER WITH THE FOLLOWING ASSUMPTIONS:

- TSI.block INSTRUCTION READS SUCCESSIVE ADDRESSES FROM SYSTEM MEMORY, DATA ELEMENT VALUES ARE 0,1,2,...etc.
- TCI.select INDEX INSTRUCTION PLACES VALUES IN PE MEMORIES USING THE FOLLOWING PARAMETERS
- ASSUME NO PE VID-to-PID TRANSLATION
- TRANSFER COUNT = 36
- PE ADDRESS = 0
- PE COUNT = 4
- LOOP CONTROL = BIP
- BASE UPDATE COUNT = 0
- BASE UPDATE = 8
- INDEX UPDATE TABLE VALUE IS 0x00EEF222 WHICH GIVES UPDATES 2,2,2,-1,-2,-2
- INDEX COUNT = 7

FIG. 19

1900

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CTU TRANSFER					I/O	TYPE		SELECT-PE				X	RSVD			CORE TRANSFER COUNT (CTC)															
RESERVED																STARTING TRANSFER ADDRESS (WITHIN PE MEMORY)															
LOOP CTRL		PE COUNT						BASE UPDATE COUNT								BASE UPDATE (STRIDE)															
INDEX COUNT (HOLD) RANGE: 1 TO 65536																RESERVED								INDEX UPDATE RANGE: 1-256							
PEMSK7				PEMSK6				PEMSK5				PEMSK4				PEMSK3				PEMSK2				PEMSK1				PEMSK0			

LOOP CTRL	LOOP CTRL SPECIFIES A PARTICULAR ORDER IN WHICH PE, BASE AND INDEX VALUES ARE UPDATED. THREE POSSIBLE ORDERS ARE SELECTABLE WHICH CORRESPOND TO THREE ASSIGNMENTS OF PE, BASE AND INDEX UPDATE TO THREE NESTED CONTROL LOOPS (OUTER, MIDDLE AND INNER). 00 - BASE (OUTER), INDEX (MIDDLE), PE (INNER) - BIP 01 - BASE (OUTER), PE (MIDDLE), INDEX (INNER) - BPI 10 - PE (OUTER), BASE (MIDDLE), INDEX (INNER) - PBI
PE COUNT	(NOT USED FOR THIS ADDRESS MODE)
BASE UPDATE (STRIDE)	DISTANCE BETWEEN SUCCESSIVE BLOCKS. UNITS ARE OF "DATA TYPE" SIZE.
BASE UPDATE COUNT	USED FOR PBI LOOP CONTROL. SPECIFIES THE NUMBER OF TIMES THE BASE IS UPDATED BEFORE EXITING TO THE OUTER LOOP (PE UPDATE). RANGE IS 1 TO 256.
INDEX COUNT (HOLD)	NUMBER OF CONTIGUOUS DATA ITEMS IN A BLOCK
INDEX UPDATE	DISTANCE BETWEEN SUCCESSIVE ITEMS WITHIN A BLOCK. UNITS ARE OF "DATA TYPE" SIZE.
PEVEC	THESE VALUES FORM A TABLE OF 4-BIT FIELDS THAT ARE USED TO SPECIFY PE SELECTIONS FOR UP TO 8 PASSES THROUGH THE PEs. FOR EACH FOUR BIT FIELD, A '1' BIT SELECTS THE PE VID CORRESPONDING TO ITS BIT POSITION. PEMSK0 MUST HAVE AT LEAST ONE '1' BIT, AND THE FIRST ALL-ZERO FIELD DETECTED CAUSES SELECTION TO BEGIN AGAIN WITH THE PEMSK0 FIELD. IN BIP AND BPI LOOP MODES, WHEN THE BASE IS UPDATED, THE PEVEC TABLE RESETS TO THE FIRST 4-BIT ENTRY REGARDLESS OF WHICH ENTRY WAS LAST IN USE. IN PBI LOOP MODE THE PEVEC ENTRIES ARE CYCLED THROUGH CONTINUOUSLY.

FIG. 20

2000

LOOP CONTROL: BIP (INDEX VARIES FIRST, THEN BASE, THEN PE ID)				
ADDRESS (WORDS)	PE0	PE1	PE2	PE3
0x0000		0	1	2
0x0001		3	4	5
0x0002	9	6	7	8
0x0003		10	11	12
0x0004				
0x0005				
0x0006				
0x0007				
0x0008		13	14	15
0x0009		16	17	18
0x000a	22	19	20	21
0x000a		23	24	25

PATTERN ABOVE RESULTS FROM AFTER A TRANSFER WITH THE FOLLOWING ASSUMPTIONS:

- TSI.block INSTRUCTION READS SUCCESSIVE ADDRESSES FROM SYSTEM MEMORY, DATA ELEMENT VALUES ARE 0,1,2,...etc.
- ASSUME PE TRANSLATE TABLE MAPS 0→1, 1→2, 2→3, 3→0
- TCI.selectpe INSTRUCTION PLACES VALUES IN PE MEMORIES USING THE FOLLOWING PARAMETERS
- TRANSFER COUNT = 26
- INITIAL PE ADDRESS OFFSET = 0
- PE COUNT = NOT USED
- LOOP CONTROL = BIP
- BASE UPDATE COUNT = 0
- BASE UPDATE = 8
- INDEX UPDATE = 1
- INDEX COUNT = 4
- PE TABLE IS 0x00000F77
  - FIRST PASS SELECT VIDs: 0, 1, 2 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3)
  - NEXT PASS SELECT VIDs 0,1,2 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3)
  - NEXT PASS SELECT VIDs 0,1,2,3 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3,0)

FIG. 21

2100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CTU TRANSFER					I/O	TYPE		SELECT-INDEX-PE				X	RSVD			CORE TRANSFER COUNT (CTC)															
IU COUNT					RESERVED											STARTING TRANSFER ADDRESS (WITHIN PE MEMORY)															
LOOP CTRL					PE COUNT			BASE UPDATE COUNT								BASE UPDATE (STRIDE)															
IU7					IU6			IU5				IU4				IU3				IU2				IU1				IU0			
PEMSK7					PEMSK6			PEMSK5				PEMSK4				PEMSK3				PEMSK2				PEMSK1				PEMSK0			

LOOP CTRL	LOOP CTRL SPECIFIES A PARTICULAR ORDER IN WHICH PE, BASE AND INDEX VALUES ARE UPDATED. THREE POSSIBLE ORDERS ARE SELECTABLE WHICH CORRESPOND TO THREE ASSIGNMENTS OF PE, BASE AND INDEX UPDATE TO THREE NESTED CONTROL LOOPS (OUTER, MIDDLE AND INNER). 00 - BASE (OUTER), INDEX (MIDDLE), PE (INNER) - BIP 01 - BASE (OUTER), PE (MIDDLE), INDEX (INNER) - BPI 10 - PE (OUTER), BASE (MIDDLE), INDEX (INNER) - PBI
PE COUNT	(NOT USED FOR THIS ADDRESS MODE)
BASE UPDATE (STRIDE)	DISTANCE BETWEEN SUCCESSIVE BLOCKS. UNITS ARE OF "DATA TYPE" SIZE.
BASE UPDATE COUNT	USED FOR PBI LOOP CONTROL. SPECIFIES THE NUMBER OF TIMES THE BASE IS UPDATED BEFORE EXITING TO THE OUTER LOOP (PE UPDATE). RANGE IS 1 TO 256.
IU COUNT	INDEX UPDATE COUNT. THIS IS THE NUMBER OF ENTRIES IN THE INDEX UPDATE TABLE. WHEN 'IU Count' INDEX UPDATES HAVE OCCURRED (WITH ASSOCIATED ACCESSES AFTER UPDATE), THE NEXT OUTER LOOP VARIABLE (B OR P) IS UPDATED. SUBSEQUENT INDEX UPDATES START AT THE FIRST ENTRY AGAIN (IU0). IF 'IU Count' IS GREATER THAN 8, THE TABLE ENTRIES ARE USED AGAIN, STARTING AT THE BEGINNING OF THE TABLE.
IUx	IU0 - IU7 FORM AN INDEX UPDATE TABLE WITH EACH ENTRY BEING A 4-BIT UPDATE VALUE. UPDATE VALUES ARE INTEGERS IN THE RANGE OF -8 TO +7.
PEMSKx	THESE VALUES FORM A TABLE OF 4-BIT FIELDS THAT ARE USED TO SPECIFY PE SELECTIONS FOR UP TO 8 PASSES THROUGH THE PEs. FOR EACH FOUR BIT FIELD, A '1' BIT SELECTS THE PE CORRESPONDING TO ITS BIT POSITION. PEMSK0 MUST HAVE AT LEAST ONE '1' BIT, AND THE FIRST ALL-ZERO FIELD DETECTED CAUSES SELECTION TO BEGIN AGAIN WITH THE PEMSK0 FIELD.

FIG. 22

2200

LOOP CONTROL: BIP (INDEX VARIES FIRST, THEN BASE, THEN PE ID)				
ADDRESS (WORDS)	PE0	PE1	PE2	PE3
0x0000		0	1	2
0x0001				
0x0002		3	4	5
0x0003				
0x0004				
0x0005	9	6	7	8
0x0006		10	11	12
0x0007				
0x0008		13	14	15
0x0009				
0x000a				
0x000a	19	16	17	18

PATTERN ABOVE RESULTS FROM AFTER A TRANSFER WITH THE FOLLOWING ASSUMPTIONS:

- TSI.block INSTRUCTION READS SUCCESSIVE ADDRESSES FROM SYSTEM MEMORY, DATA ELEMENT VALUES ARE 0,1,2,...etc.
- ASSUME PE TRANSLATE TABLE MAPS 0→1, 1→2, 2→3, 3→0
- TCI.selectpe INSTRUCTION PLACES VALUES IN PE MEMORIES USING THE FOLLOWING PARAMETERS
- TRANSFER COUNT = 20
- INITIAL PE ADDRESS OFFSET = 0
- PE COUNT = NOT USED
- LOOP CONTROL = BIP
- BASE UPDATE COUNT = 0
- BASE UPDATE = 6
- INDEX COUNT = 3
- INDEX TABLE = 0x00000032 (+2, THEN +3)
- PE HELPE IS 0x00000F77
  - FIRST PASS SELECT VIDs 0,1,2 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3)
  - NEXT PASS SELECT VIDs 0,1,2 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3)
  - NEXT PASS SELECT VIDs 0,1,2,3 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3,0)